N94- 32432

CONFIG - INTEGRATED ENGINEERING OF SYSTEMS AND THEIR OPERATION

Jane T. Malin
Automation & Robotics Division, Engineering
NASA Johnson Space Center
Houston, TX 77058

Dan Ryan
MITRE
Houston, TX 77058

Land Fleming
Lockheed
Houston, TX 77058

S12-61
24175
p 8

ABSTRACT

This article discusses CONFIG 3, a prototype software tool that supports integrated conceptual design evaluation from early in the product life cycle, by supporting isolated or integrated modeling, simulation, and analysis of the function, structure, behavior, failures and operation of system designs. Integration and reuse of models is supported in an object-oriented environment providing capabilities for graph analysis and discrete event simulation. CONFIG supports integration among diverse modeling approaches (component view, configuration or flow path view, and procedure view) and diverse simulation and analysis approaches. CONFIG is designed to support integrated engineering in diverse design domains, including mechanical and electro-mechanical systems, distributed computer systems, and chemical processing and transport systems.

INTRODUCTION

The core of engineering design and evaluation focuses on analysis of physical design. Today's computer-Aided Engineering (CAE) and Product Data Management (PDM) software packages can support concurrent engineering, bringing together engineering and production design. However, depending on the engineering domain, they are oriented toward geometry or continuous process and control parameters. They do not provide enough support for conceptual design early in the life cycle or for engineering for operation, fault management, or supportability (reliability and maintainability). Integrated modeling and analysis of system function, structure, behavior, failures and operation is needed, early in the life cycle. The same models can also be reused to support design of fault management software and procedures for the system.

Benefits of concurrent engineering include reduced costs and shortened time for system development. Benefits of engineering for operations and supportability include more robust systems that meet customer needs better and that are easier to operate, maintain and repair. Benefits of reuse in the design of software and procedures include faster software development and more robust fault management.

Conventional system modeling approaches were not designed for evaluating conceptual designs early in the system life cycle. These modeling approaches require more knowledge of geometric or performance parameters than is usually available early in design. Thus, designers rely on "engineering judgment" or systems engineering analysis for early design evaluation. Usually, there is not a traceable path from these analyses to the conventional simulations that are done later. Also, these conventional simulations are often too special-purpose to support evaluations of operability, diagnosability, and supportability.

A more abstracted level of modeling would be sufficient for early conceptual design definition and evaluation, and would also remain useful for some later analyses. Component-connection models are one such useful abstraction. Discrete event models are another useful abstraction. Discrete event simulation technology combines both abstractions, and has been used extensively for evaluation of conceptual designs of equipment configurations in operations research (3). In CONFIG, these abstractions, with some enhancements, are also proving to be useful in defining and evaluating conceptual designs for several types of systems.

Design Goals of CONFIG

When the CONFIG project began, the goal was to support simulation studies for design of automated diagnostic software for new life-support systems (9). The problem was to design an "expert system" on-line troubleshooter before there was enough experience with the system for there to be an expert. The design engineer could use a model of the system to support what-if analyses concerning how failures would propagate, interact, and become observable and testable. With these analyses, development of detection and diagnosis procedures could proceed in parallel with system design. This activity is similar to Failure Modes and Effects Analysis (5), but with comparative simulations of failure effects for the purpose of developing diagnostic software. Conventional simulation software was not up to this challenge, but discrete event simulation software, with enhancements, seemed promising. CONFIG supports the use of qualitative models for applying discrete event simulation to continuous systems, and the use of graph analysis on component-connection models.

CONFIG is designed to model many types of systems in which discrete and continuous processes occur. The CONFIG 2 prototype was used to model and analyze effects of failures in: 1) a simple two-phase thermal control system based on a Space Station prototype thermal bus, 2) a reconfigurable computer network with alternate communications protocols, and 3) Space Shuttle Remote Manipulator System latching and deployment subsystems (7). The core ideas of CONFIG have been patented (8). CONFIG 3 has added capabilities for graph analysis and for modeling operations and procedures. Many potential uses for CONFIG in engineering and operations have been identified.

How CONFIG Can Support Engineering Activities

Major engineering activities include systems engineering (functions), physical design engineering (materials, processes, equipment performance and geometry, manufacturing, etc.), operations engineering (control, diagnostics, procedures), and supportability (reliability, maintainability) and safety (failures and hazards) engineering. There has been much progress in tool development to support systems engineering, physical design engineering, and control engineering. However, there have been missing links between physical design engineering and the other engineering areas. There has also been little software support for operations, supportability and safety engineering. Major design goals of the CONFIG tool include support for conceptual design for operations and safety engineering, and for bridging the gaps between physical design engineering and other types of engineering. These types of support will be discussed in the sections that follow in the paper.

A major area of potential CONFIG 3 use is for concurrent or integrated engineering that addresses operability, diagnosability and supportability. Operations procedures and software can be modeled along with the system model, to evaluate specific procedures, protocols, rules or plans, when they are applied in various system configurations and scenarios. Simulations of components and their interactions during operations can show how the system realizes functional requirements. Functional requirements can be embodied in operations models. Approaches to sensor location can be evaluated for support of test and diagnosis requirements. Redundancy, observability and composite measurements can be investigated in operations scenarios and fault management scenarios.

Another major area of potential CONFIG 3 use is for design definition, evaluation and documentation. Component-connection models are well suited for configuration and connectivity analysis, and for analysis of potential interactions among system parts. Graph analyses can be used to investigate completeness, consistency, modularity, efficiency, redundancy, fault tolerance and criticality. Graph analyses can also be used to locate potential failure impacts and sources. Discrete event simulation is well suited for predicting critical system states in nominal operation or when one or more failures occur. Such simulations can be used to compare alternative designs to each other and to specifications, and may include statistical studies. For design documentation, CONFIG can be used to capture functional goals in component and operations models.

CONCEPTUAL DESIGN SUPPORT

A major goal of the CONFIG project is to support conceptual design for operations and safety engineering. Major tasks in conceptual design are design definition, evaluation (by simulation and analysis) and documentation. In operations engineering, the focus is on the design of systems and procedures for operating, controlling and managing the system in normal or faulty conditions. In safety engineering, the focus is on prevention of hazardous effects and conditions in the physical system or its operation. In these types of engineering, complex interactions and interfaces among system components and operations must be a focus. This is so because of interest in such issues as operations efficiency and completeness, redundancy, fault tolerance and diagnosability. Yet, conventional physical simulation modeling provides quantitative and geometric detail where it is not needed, and leaves out modeling that is needed of operations, faults, and component interactions. Systems engineers can model functional components and their connections, but are not supported in modeling a corresponding physical design and operations design that together achieve these functions.

Component-connection representations are well suited for modeling and defining both physical system designs (as structures of interacting components) and operations designs (as structures of interacting actions), as well as the interactions between system components and operational actions. Discrete event models have been used for this type of modeling in areas that focus on queueing and scheduling problems, but can be extended to support conceptual modeling in operations and safety engineering. This type of modeling is also compatible with systems engineering function diagrams (1).

When executable component-connection models define conceptual designs, they provide rich design documentation that can achieve most design knowledge capture goals. Design decisions can be specified by modeling operations and systems jointly, thereby showing functional intent. Design alternatives can be documented as executable models and their evaluation results. Functional intent can also be explicitly documented in operations activity models.

Discrete event simulation technology can be adapted for this purpose by incorporating abstracted modeling of component behavior and operating procedures. Such enhancements should accommodate modeling of both nominal and faulty component operating modes and both nominal and recovery operations. These modeling capabilities can be achieved by applying process modeling approaches from qualitative modeling and plan representation approaches from planning in artificial intelligence research.

The principal types of simulation and analysis enhancements needed are global capabilities, since the discrete event simulation approach is limited to local propagation of discrete change events through a component-connection structure. These global capabilities can be achieved by applying graph analysis techniques to the global component-connection structure during simulation, and in static analyses.

INTEGRATING TYPES OF ENGINEERING

Another goal of the CONFIG project is to help bridge the gaps between physical design engineering and other types of engineering, especially operations engineering and systems engineering. Such gaps impede progress in integrating engineering. The most important cause of these gaps is lack of modeling approaches that could support clean mappings among models. The CONFIG project integration philosophy is support for loose coupling among engineering support tools, so that relevant information from one tool can be selected, dispatched and mapped or translated for use by another tool. Although a single data base of interrelated models is not needed for integration, mappable modular information is needed from each tool, and distributed operation of heterogeneous tools should be assumed. Attention also needs to be paid to how selected information can be reused and how consistency can be maintained. Since many such tools, including CONFIG, use model libraries, such issues can be dealt with at both the library and the application level.

Some of these issues have been addressed in CONFIG design, to support internal integration of modeling approaches and separation of modeling and analysis concerns. Since CONFIG is intended to support incremental modeling and modeling options, modularity of models and analyses is supported. This modularity is supported by the object-oriented approach and explicit interface definitions among model types and between models and their graphical presentations.

To integrate with other types of engineering, relations between notations and representations need to be identified. Such relations can form a basis for the activities of selection, translation and mapping that support coordinated and integrated engineering. To illustrate integration with physical design engineering, we use a process engineering example (11). In process engineering, component-connection models can correspond to the process plant structures that are used in the sequential-modular approach to generating flowsheeting simulations. Likewise, components correspond to equipment, which is modeled next.

Component-connection models are closely related to functional diagram notation in systems engineering. In addition, functional information, in the form of goals for actions, is central to operations models in CONFIG. These goals describe variables that correspond to states in the system model components. Mapping to systems engineering models should not be difficult.

In operations and support engineering, tools are emerging for scheduling, planning, and representing procedures. The integration problem is to support conceptual design of operations so that the results can be reused in these tools. Operations modeling in CONFIG has been designed to correspond to current planning representations, by including goals in the activity models to achieve or maintain states. Thus, for example, CONFIG could be used both to evaluate the output of a planning or procedure development tool, and to provide a problem to a planning tool. These same capabilities could be used during operations, for evaluation of design changes for procedures, plans or schedules in the context of the current situation.

CONFIG 3

The focus of CONFIG 3 work has been on preparing and demonstrating a solid foundation for both product development and further integration studies. CONFIG has been reimplemented in a standard object-oriented language, in modular and well-documented form. The project approach has been to incrementally integrate advanced modeling and analysis technology with more conventional technology. The prototype integrates qualitative modeling, discrete event simulation and directed graph analysis technologies for use in analyzing normal and faulty behaviors of dynamic systems and their operations.

CONFIG 3 has been designed for modularity, portability and extensibility. A generic directed graph element design has been used to standardize model element designs and to promote extensibility. This directed graph framework supports integration of levels of modeling abstraction and integration of alternative types of model elements.

CONFIG provides intelligent automation to support nonprogrammer and nonspecialist use and understanding. CONFIG embeds object-oriented model libraries in an easy-to-use toolkit with interactive graphics and automatic programming.

Enhanced Discrete Event Simulation Capabilities

In traditional discrete event modeling and simulation, state changes in a system's entities, "events", occur discretely rather than continuously, and occur at nonuniform intervals of time. Throughout simulation, new events are added to an event list that contains records of events and the times they are scheduled to occur. Simulation processing jumps from one event to the next, rather than occurring at a regular time interval. Computation that results in creation of new events is localized in components, which are connected in a network. Any simulation run produces a particular history of the states of the system, and random variables are used in repeated runs to produce distributions of system output variables. These statistical simulation experiments are used to compare design alternatives.

To enhance this discrete event simulation approach to accommodate abstracted qualitative modeling of continuous behavior of system components, a number of new concepts and methods were developed for CONFIG. These concepts and methods include a component model with operating modes, types of links connecting components ("relations" and "variable clusters"), new state transition structures ("processes"), methods for representing qualitative and quantitative functions ("process language"), and a new simulation control approach.

These enhancements make discrete event simulation techniques available for evaluating conceptual designs for systems and their operations. Engineers can investigate how system components will interact in operations scenarios, in which some components can be nominal and some can be faulty, and in which effects of single or multiple faults can be local or can interact and propagate through the system. Simulations can be used to see whether system designs meet functional or redundancy requirements. Simulations can also be used to investigate alternatives for instrumenting the system, and for detecting and diagnosing faults.

Digraph Analysis Capabilities

The CONFIG Digraph Analyzer (DGA) makes graph analysis techniques available for evaluating conceptual designs of systems and their operations. The DGA is based on reachability search. This search is implemented generically so that it can be applied to any of the many types of graph data structures in CONFIG. The DGA user may specify constraints that limit the search in various ways. The results of the reachability analysis may be written to a file, presented as a textual display, or the paths found may be highlighted on an iconic screen display of the graph. In textual output mode, the DGA may also display metrics associated with the graph topology such as path lengths. Since the DGA is generic, it can be used on simplified component-connection models of systems or operations before detailed modeling has been completed.

The ability to impose constraints on the graph search allows the user to tailor analyses for a wide range of purposes. Analyses of completeness, consistency and modularity can be supported, such as ensuring that all electrically powered devices in a model are on an electrical circuit. Analysis of failure sources and impacts can be done by tracing the paths of impact of a given failure source.

System Modeling

Devices are the basic components of a CONFIG system model, which are connected together in topological model structures with Relations. Device behavior is defined in operating and failure Modes, which contain mode dependent and mode transition processes. Modes are connected together in a mode transition diagram which delineates the transition dependencies among the individual modes. Device Processes define change events in device variables, which are conditionally invoked and executed with appropriate delays during a simulation. In terms of qualitative process theory (4), a change in a component variable or a mode can be equivalent to passing a landmark value and reaching a new qualitative range. Processes define time-related behavioral effects of changes to device input variables, both direction of change and the new discrete value that will be reached, possibly after a delay. Faults and failures can be modeled in two distinctly different ways. Failure modes can be used to model device faults. Mode-transition processes can be used to model device failures that cause unintended mode changes.

Relations connect devices via their variables, so that state changes can propagate along these relations during simulations. Related variables are organized into variable clusters, to separate types of relations by domain (e.g., electrical vs. fluid connections). Relations can also connect Devices with device-controlling Activities in operations models.

Flow Path Modeling

There are two inherent difficulties in modeling flows by means of CONFIG device processes. First, processes in CONFIG are by definition local descriptions of a device's behavior while flow is in fact a global property of the modeled system and the substances being subjected to flow within the system. Second, in many cases a modeled system can undergo dynamic changes in topological structure during the course of its operations, while any process descriptions involving flow must often rely on assumptions that some aspects of the device's relationship to the system topology are static. These factors severely limit the reusability of device descriptions to a limited set of possible system topological structures.

A flow-path management module (FPMM) was implemented to address these problems, by interrupting simulation to operate on a global flow path layer of the model. The FPMM is separate from the module implementing local device behavior, but the two modules are interfaced via flow-related state variables in the modeled system's component devices. During simulation, the behavior modeling module notifies FPMM of local changes in a device's state and FPMM recomputes the global effects on flows produced by the local state change. The FPMM then updates the state of flow in all device affected by the recomputation, and this in turn may cause other processes to be invoked that result in further local changes. This interface design makes it possible for the user to write local device process descriptions that do not depend on any assumptions concerning the system topology but yet are capable of describing the mutual dependencies between the device and the system flows. Therefore, these process descriptions are highly reusable.

For large models, it would not be feasible to examine each device in a system every time any one of them underwent a process-induced state change during a simulation. FPMM therefore constructs a simplified representation of the system as a collection of aggregate objects referred to as "circuits." The devices within a given circuit are not manipulated by FPMM unless the flow state of the circuit has changed.

The complexity of the algorithm for processing the effects on system flows due to one device state change is the product of the average number of devices per circuit times the average number of circuits per device. Unfortunately, the number of circuits is itself a nonpolynomial function of the average number of connections to a device in the system (degree of the node). To increase the size of systems for which the algorithm's complexity is tractable, a second class of aggregates referred to as "clusters" was introduced. Clustering reduces a graph to a hierarchy of alternating serial and parallel clusters (6). The complexity of flow computations is a linear function of the average degree of the nodes for cluster-based representations. There are many practical examples of systems that are only partially reducible to clustering representations. For such systems, FPMM produces a hybrid representation consisting of a set of circuit objects in which clusters are treated as individual nodes. This hybrid approach can represent any arbitrary system topology, unlike pure clustering, and will allow considerably more efficient flow computations for most topologies than would pure circuit representations.

The "circuits", which are component configuration representations, simplify and separate analysis of state changes that produce changes in global configuration of flow. In many cases, configuration determination alone can be sufficient to verify flow/effort path designs, to establish flow paths for a continuous simulation, for reconfiguration planning, and for troubleshooting analysis (see Ref. [2] on cluster-based design of procedures for diagnosis, test, repair and work-around in a faulty system).

Operations modeling

Activities are the basic components of a CONFIG operations model, which are connected together in action structures with Relations. They represent procedures or protocols that interact with the system, to control and use it to achieve goals or functions. Each activity model can include evaluable specifications for what it is intended to achieve or maintain. Activity behavior is defined and controlled in a sequence of phases, ending in an evaluation of results. Activity behavior is defined by processes that simply model direct effects of actions, or that control device operation and mode transitions to achieve activity goals. Relations define sequencing and control between activities and connect Devices with device-controlling Activities.

Operations models are designed to support operation analysis with procedure models. These models are designed to support analysis of plans and procedures for nominal operation. They are also designed to support simulation and analysis of proposed design changes (reconfigured systems and revised procedures) that are developed during operations in response to failures. The procedure modeling elements are designed for reuse by intelligent replanning software, and for compatibility with functional modeling in systems engineering.

Model Development & Integration Capabilities and Approach

CONFIG provides extensive support for three separable yet tightly integrated phases of user operation during a modeling session: Library Design, Model Building, and Simulation and Analysis. This includes a graphical user interface for automated support of modeling during each of the phases including the development of object-oriented library element classes or templates, the construction of models from these library items, model inspection and verification, and running simulations and analyses.

The integration between the phases enables an incremental approach to the modeling process by allowing the user to repeatedly and rapidly incorporate into a perhaps initially simple model, lessons learned from a simulation or other analysis of that model or combination of models. This information might be used to explore a range of progressively more detailed, or simply different, structural, behavioral, and functional modifications to the various types of CONFIG models such as: addition, deletion, and reconfiguration of elements of the model layout; substitution of functionally similar elements; selective modification and redefinition of the structure and behavior of the classes from which the elements in a model were instantiated. Additionally, CONFIG's support for these phases as separate user activities fosters the achievement of concurrent engineering goals by allowing library definition, model building, and model analysis to be performed by different individuals at different times depending on area of expertise and availability of resources. Finally, support for the model building phases spans all types of modeling that can be performed in CONFIG including component structure, behavior and flow, and activity goals and structure.

The various types of model elements are instances of class definitions which are located in libraries, either CONFIG-provided or user-defined. These libraries are themselves objects which are hierarchically organized to utilize the full benefits of the object-oriented approach including inheritance. The library builder may construct one library by accessing the elements of the superlibrary and creating new elements as subclasses of the superlibrary's elements. Thus, a fundamental aspect of the automated modeling process in CONFIG is its support for creating, modifying, and storing libraries of classes, or model element templates, from which models may be constructed.

Initially the library designer may create his own object-oriented and extensible Qualitative Process Language of Variable Types, Operators, and specialized Operations which will subsequently be used to describe Device level behavior. This ability to define a domain-specific language is an important feature of CONFIG in that it does not restrict the modeler to some particular modeling language and allows him to describe the system easily using his own qualitative or quantitative vocabulary. Additionally, the library designer may then go on to create specialized subclass hierarchies of a number of other different types of CONFIG elements including Devices, Relations, Variable Clusters, and Activities.

The Device Class hierarchy illustrates how CONFIG uses the power of object-oriented definition of model elements. A Device Class defines a template from which a specific qualitative model of device behavior may be instantiated or stamped out. Device Classes not only define attributes or variables for which all instances of the class will provide instance-specific values, but also Modes of operation consisting of Mode Dependent and Mode Transition Processes that are conditionally invoked and executed with appropriate delays during a simulation. The various modes of operation are grouped together in the Mode Transition Digraph, a composite object, which may be incrementally modified as it is inherited down the Device Class hierarchy. The Processes associated with Modes contain Statements which are written by the modeler making use of the domain-specific Qualitative Process Language.

Creation of various configurations of system models may then proceed graphically and interactively. The model builder makes use of a model building window in which instances of Device and Relation Classes from a particular library are selected via mouse interaction from palettes. These instances are connected and arranged on a design canvas. Finally, CONFIG's graphical support for graph reachability and flow analysis and the running of simulations may then be used. Digraph analysis may be used on models with simpler device behavior definitions, and discrete event simulation can use more detailed device models.

Although CONFIG has been designed as a complete object-oriented qualitative modeling and simulation system which is able to stand alone with communication proceeding through use of its graphical user interface, an additional goal has been to provide the ability to integrate CONFIG with other CAE tools and data base management systems. This goal has been achieved and is reflected in the implementation of the CONFIG Programmatic Interface (CPI), which defines a set of functions and protocols for interacting with CONFIG. The CPI provides an avenue for other systems to interact with CONFIG, and for integration with object-oriented data bases.

Hosting

Care was taken to select software platforms for CONFIG that are portable to most Unix work stations. The Common Lisp Object System (CLOS) was selected as the software platform for this reason. CLOS is a highly standardized language, and two vendors produce Lisp compilers for most of the commonly available work stations. The user interface was implemented using Common Lisp Interface Manager (CLIM), another standardized tool built on CLOS and available from the same vendors. The most recent version of CLIM is designed to exploit the resources of the particular windowing system being run by the host machine so that the "look and feel" of CONFIG can be familiar.

Areas of future work

There are several areas for enhancement that we are planning to pursue. One is to provide a more complex and complete operations modeling capability. Another is to further enhance the discrete event simulation capabilities to include facilities for managing and documenting simulation experiments. We plan to integrate with an object-oriented data base management system. We also have plans for improving and enhancing flow path management and digraph analysis capabilities.

CONCLUSIONS

The CONFIG prototype demonstrates advanced integrated modeling, simulation and analysis to support integrated and coordinated engineering. CONFIG supports qualitative and symbolic modeling, for early conceptual design. System models are component structure models with operating modes, with embedded time-related behavior models. CONFIG supports failure modeling and modeling of state or configuration changes that result in dynamic changes in dependencies among components. Operations and procedure models are activity structure models that interact with system models. The models support simulation and analysis both of monitoring and diagnosis systems and of operation itself. CONFIG is designed to support evaluation of system operability, diagnosability and fault tolerance, and analysis of the development of system effects of problems over time, including faults, failures, and procedural or environmental difficulties.

CONFIG has many attributes that aid commercialization. The core CONFIG concepts are patented. CONFIG integrates advanced technology with mature discrete event simulation and digraph analysis technology bases. In the years of work on CONFIG, a number of requirements have been discovered and a number of technical and product problems have been solved. The prototype and its design are well documented, to ease conversion of all or part of the design to a supported product. CONFIG provides hooks and placeholders for further enhancements. CONFIG takes advantage of sophisticated capabilities in object-oriented databases and graphical interfaces. Commercial versions of these technologies appear to be mature enough now to support this type of advanced CAE tool.

There are several possible commercialization approaches for CONFIG. One is to simply develop a commercial version of the CONFIG tool. Another is to enhance an existing tool for object-oriented modeling or discrete-event simulation. Another approach is to integrate CONFIG with a Process simulation or Control engineering tool in a CAE environment.

ACKNOWLEDGEMENTS

The authors wish to thank Bryan Basham for significant design and coding contributions to all aspects of the CONFIG 3 prototype, prior to his leaving the project. We also thank Leslie Ambrose, Ralph Krog and Debra Schreckenghost for their contributions to user interface design, Brian Cox for his contributions to discrete event simulation design, Daniel Leifker, for his contributions to operations modeling design, and Sherry Land, for her contributions to digraph analysis design. We also thank Kathy Jurica for her continuing management support.

REFERENCES

1. Alford, M. Strengthening the System Engineering Process, Engineering Management Journal, Vol. 4, No. 1, March, 1992 , pp 7-14.

2. Farley, A. M. Cluster-based Representation of Hydraulic Systems, Proc. 4th Conference on AI Applications, March, 1988, pp. 358-364.

3. Fishman, G. S. Principles of Discrete Event Simulation. New York, NY: Wiley, 1978.

4. Forbus, K. Qualitative Physics: Past, Present, and Future. In Exploring Artificial Intelligence (H. Shrobe and AAAI, eds.). San Mateo, CA: Morgan Kaufmann, 1988.

5. Fullwood, R. R. and Hall, R. E. Probabilistic Risk Assessment in the Nuclear Power Industry: Fundamentals and Applications. Pergamon Press, 1988.

6. Liu, Z. and Farley, A. M. "Structural Aggregation in Common-Sense Reasoning". Proc. 9th National Conference on Artificial Intelligence (AAAI-91), July , 1991, pp. 868-873.

7. Malin, J. T., B. D. Basham and R. A. Harris, "Use of Qualitative Models in Discrete Event Simulation for Analysis of Malfunctions in Continuous Processing Systems." Artificial Intelligence in Process Engineering (M. Mavrovouniotis, ed.), Academic Press, pp. 37-79, 1990.

8. Malin et al., U. S. Patent 4,965,743, "Discrete Event Simulation Tool for Analysis of Qualitative Models of Continuous Processing Systems" October, 1990.

9. Malin, J. T., and Lance, N. "Processes in construction of failure management expert systems from device design information". IEEE Trans. on Systems, Man, and Cybernetics, 1987, SMC-17, 956-967.

10. Malin, J. T. and Leifker, D. B. "Functional Modeling with Goal-Oriented Activities for Analysis of Effects of Failures on Functions and Operations". Informatics and Telematics, 1991, 8(4), pp 353-364.

11. Winter, P. Computer-Aided Process Engineering: The Evolution Continues. Chemical Engineering Progress, February, 1992, pp 76-83.